Retro Computing

A Personal Journey

crude state of affairs of the computer field when I was growing up.

Since I was involved with the computer industry going back to the 1970s, I enjoy recollecting my experiences.

I have lately become fascinated by what seems to be a growing interest in the "retro" computing hobby, so there are some mentions on this page about projects recently initiated by others. I have built simulators for the DEC PDP-11/70, the DEC PDP-10, and the DEC PDP-1 computers.

Early Logic Experiences

My first exposure to anything programmable was the Z-Man toy I got when I was about 7. The Z-Man was a motorized robot with a "brain" which consisted of a circular copper PC board. Arranged around the board were a series of spring-loaded switches. A rotor driven from the wheels moved around the PC board and depending on each switch position, the two front drive motors would either steer the Z-Man left or right. Therefore, he could be "programmed" to follow a course. He even had taillights that would illuminate left or right depending on which motor was energized.

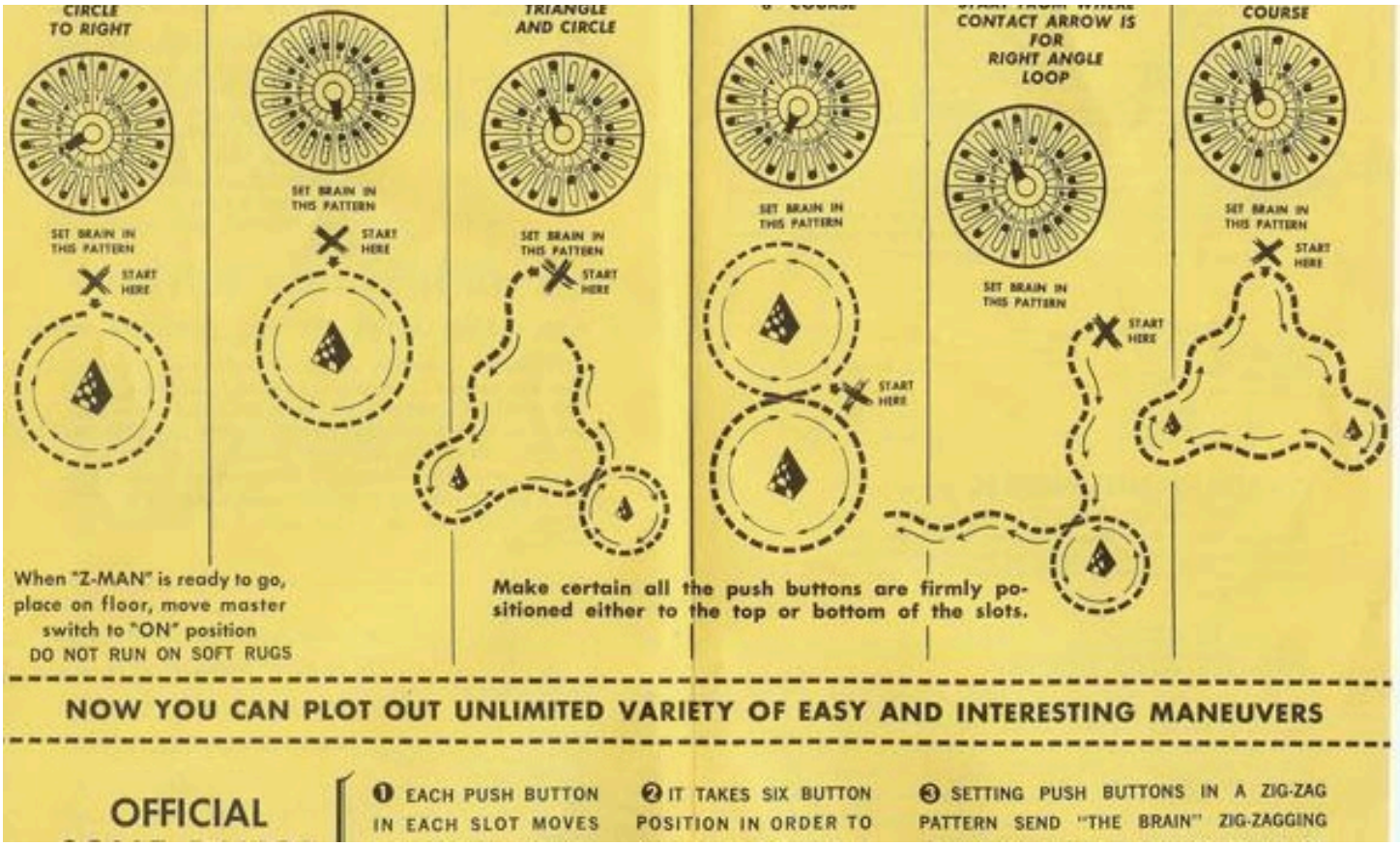
This was a very clever toy, but I was more interested in how it worked than playing with it.

It's New!
THE ELECTRONIC "BRAIN"
MANEUVERS IN OVER 400 PATTERNS
You control the man-of-the-future . . see him follow any course you plot, fire missiles, dodge obstacles . . all at your command!
ONLY
$10.95
Missiles ready for firing

The Z-Man ca. 1956 - about $120 in 2023 $s

This was the "brain" showing the switches and the programming instructions. Z-Man also sported 2 missiles which were launched when the yellow dials (also driven from the wheels) rotated around to trigger their spring-loaded releases.

This could hardly be considered a computer, but it did illustrate the concept of a stored program, which planted a seed for me that you could make things behave in a pre-determined manner at will.


CIRCLE TO RIGHT
SET BRAIN IN THIS PATTERN
START HERE
SET BRAIN IN THIS PATTERN
START HERE
TRIANGLE AND CIRCLE
SET BRAIN IN THIS PATTERN
START HERE
SET BRAIN IN THIS PATTERN
START HERE
CONTACT ARROW IS FOR RIGHT ANGLE LOOP
SET BRAIN IN THIS PATTERN
START HERE
COURSE
SET BRAIN IN THIS PATTERN
START HERE
When "Z-MAN" is ready to go, place on floor, move master switch to "ON" position
DO NOT RUN ON SOFT RUGS
Make certain all the push buttons are firmly positioned either to the top or bottom of the slots.
NOW YOU CAN PLOT OUT UNLIMITED VARIETY OF EASY AND INTERESTING MANEUVERS
OFFICIAL
❶ EACH PUSH BUTTON IN EACH SLOT MOVES
❷ IT TAKES SIX BUTTON POSITION IN ORDER TO
❸ SETTING PUSH BUTTONS IN A ZIG-ZAG PATTERN SEND "THE BRAIN" ZIG-ZAGGING

In middle school, we were subjected to the "new math" and we had textbooks from Yale University that looked like they were produced on typewriters, along with crudely drawn diagrams (probably by a bunch of grad students). The good part of this (and maybe the only good part) was an introduction to set theory and symbolic logic, which I found interesting. However, there were no practical ways for me to use that knowledge until much later. When I learned about Boolean algebra, later on, the symbolic stuff made sense.

Minivac 601

In high school, a friend had a Minivac, which was an electromechanical "computer" designed by Claude Shannon, the famous information theorist and technologist. It was sold by Scientific Development Corporation. The rotary dial was motor-driven to provide sequences of events and the relays could be used to store 1s and 0s. You could wire up patch cords to make the thing do different logical operations. I fiddled with it a bit but was not charmed by the very limited relay-based logic.

"digital computer" and cost $64.95. In 2025 dollars, that's about $670, and I am amazed my parents sprang for it. I know they wanted to encourage my interest in technology, even though they likely had no idea what this thing was.

The NORDAC consisted of a silk-screened panel with images of 10 logic gates, all of them simple 4-input NOR (hence the name NORdac) gates. (You can build any kind of logic circuit with enough NOR gates, although that is not the optimal way to do it). Rivets were used for input and output connection points and you wired the gates together with patch cords pushed into the rivets. There were 5 switches for input and lights for outputs. The kit required soldering germanium transistors to the rear of the panel, and resistors (making it an implementation of resistor-transistor logic). Power was supplied by a three-terminal 45-volt battery. One of the voltages supplied an opposing polarity bias for the bases of the transistors so that they would turn off completely with no input voltage.

In any event, I was able to fill in boolean algebra truth tables and see the results on the trainer. Even things like simple flip-flops and registers could be wired up. You could also build an adder and shift register circuits. You were able to quickly learn Boolean algebra and binary arithmetic and see things in action. It was very cool. Once I was very familiar with binary logic, the trainer was no longer of interest to me, so I donated it to the high school physics department (the closest thing to a computer science discipline we had at the time). The things I learned from that kit gave me a huge step up on what eventually became my chosen career in computers since I never forgot my self-taught lessons in Boolean logic.

Here is the actual ad that triggered my interest in the NORDAC in 1962! I am sure I used the order form from Popular Science (I had a subscription) to place the order, with my dad writing the check.

BUILD an Electronic Digital COMPUTER for $64.95

NORDAC . . . the only fully-transistorized electronic digital Computer Kit available at any price ■ Demonstrates all basic operations of million dollar commercial computers ■ Manuals provide introduction to computer

terminals and connectors ■ High speed performance . . . 50 thousand operations per second ■ High Reliability . . . constructed using professional computer components.

The NORDAC is designed for those curious about the world of electronic computers. With The NORDAC and accompanying manuals you can explore all basic operations of modern digital computers, understand basic computer concepts, principles and techniques. **$64.95**

FREIGHT COLLECT,
WATERTOWN, MASS.

Satisfaction Guaranteed

NO KNOWLEDGE OF ELECTRONICS OR ADVANCED MATHEMATICS NECESSARY

SCIENTIFIC DEVELOPMENT CORPORATION

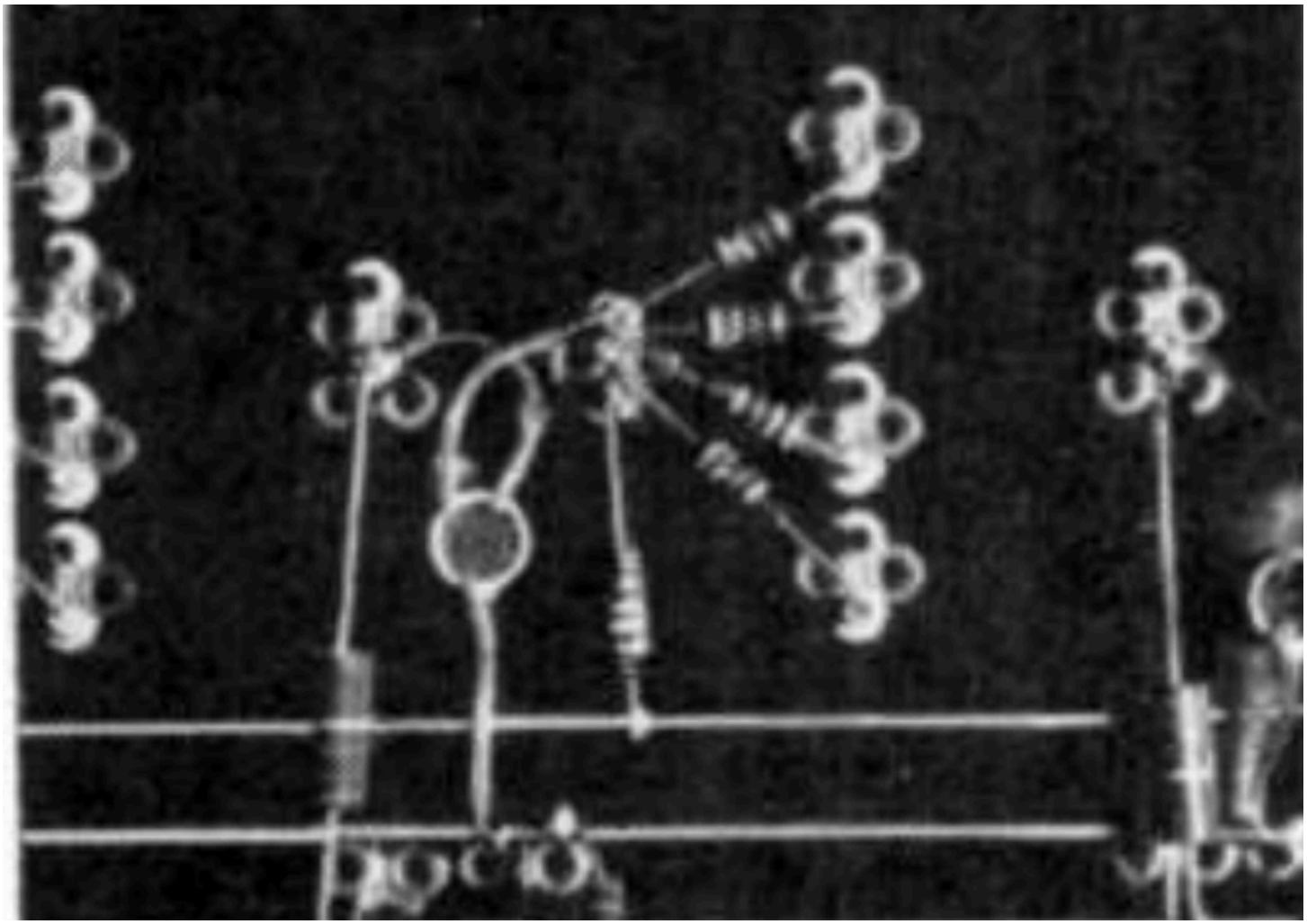
372 Main Street, Watertown, Massachusetts
Attention: Mr. B. Regan

☐ Please send free information on NORDAC.
☐ Please send ..... NORDAC Computer Kit(s).
☐ Check for $..... enclosed.
☐ Ship my NORDAC C.O.D.

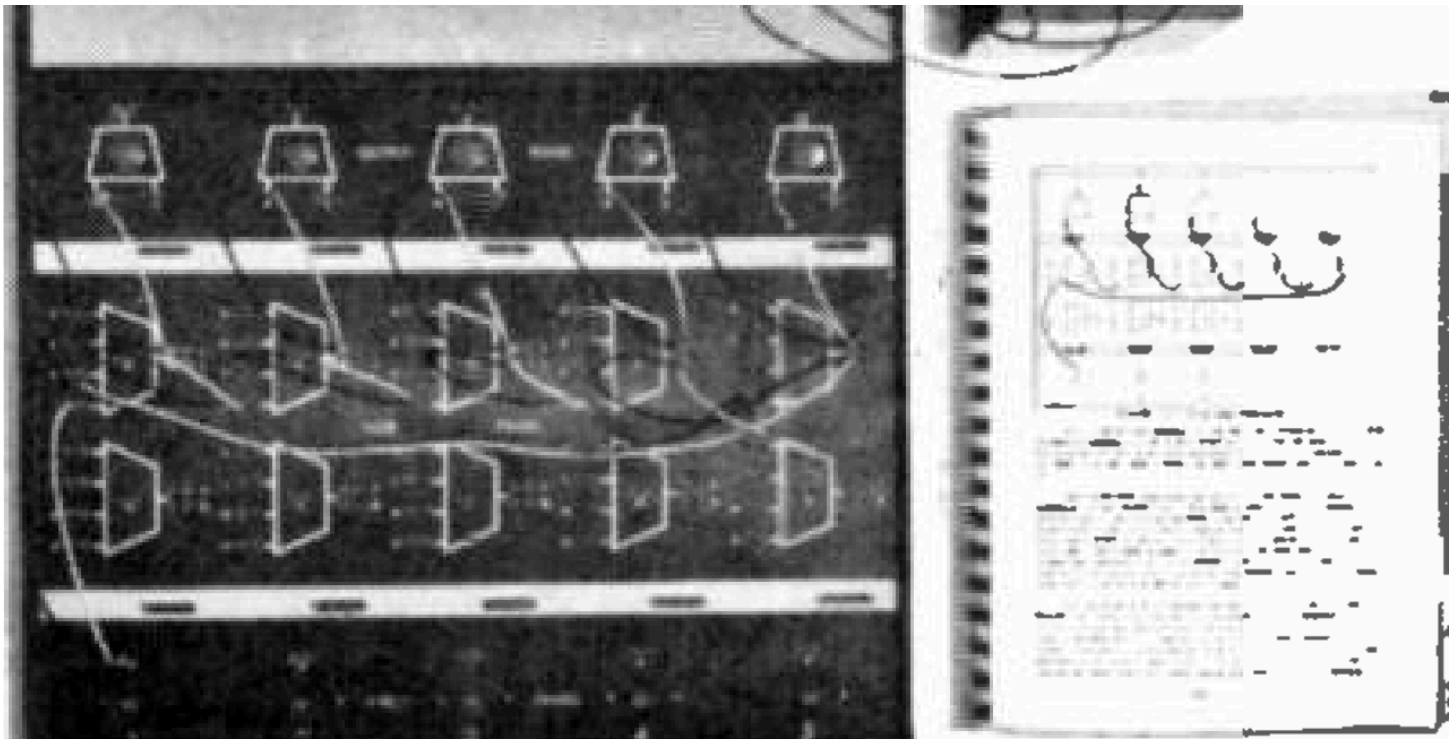
Name.....

Address:.....

City.....Zone.....State.....



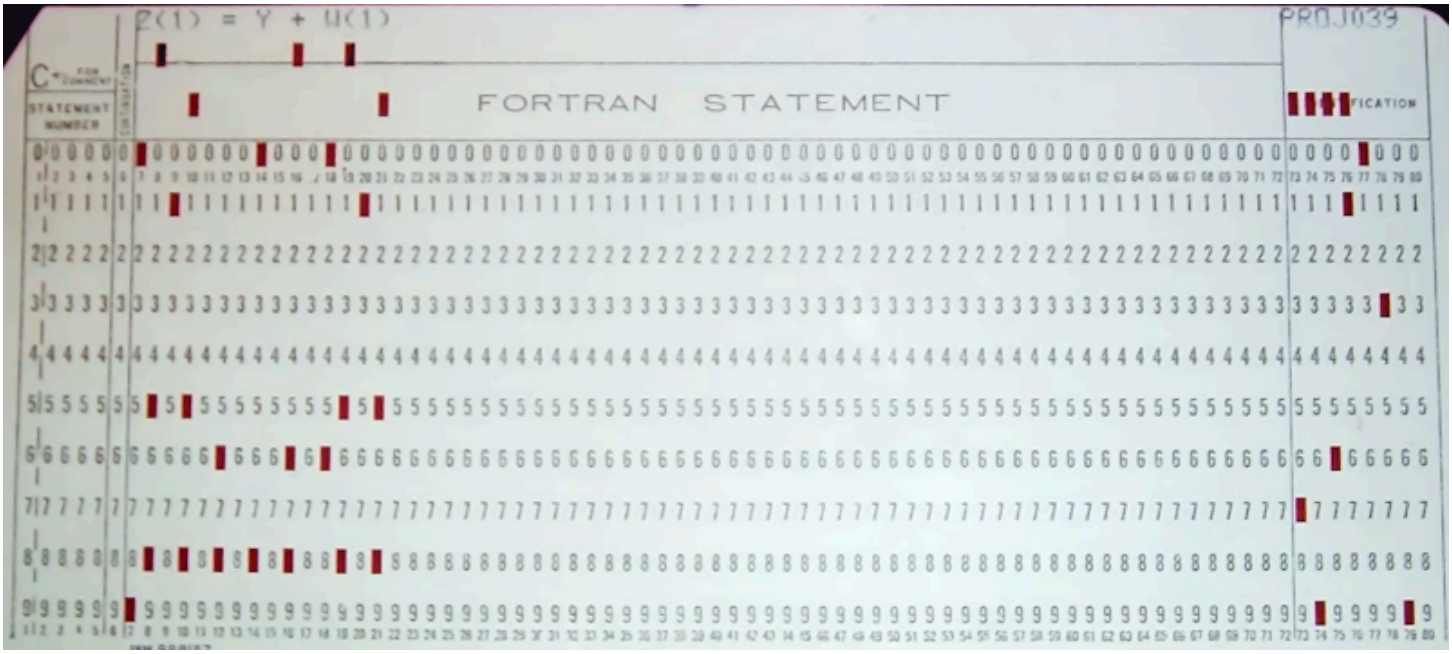
This is what the wiring on the back of the panel looked like for one gate, The bus bars supplied battery voltage across the back of the panel. Also shown is the front panel with some patch cords, the battery, and the manual. I found these photos in another advertisement in the April 1963 edition of Popular Electronics Magazine.



Speaking of High School, I recently learned of the passing of a classmate, John Walker. John was a soft-spoken, private guy who became the founder of Autodesk and the creator of AutoCAD. John and I were not close, but I am amazed at his career and accomplishments. He stepped down as CEO and moved to Switzerland in the 1980s, continuing to develop software for the company and exploring his other extensive interests. Please take a look at his personal website here.

Programming

universities offering TS services. As I recall, the college had an IBM 1130 "mainframe" and you had to submit your programs on punched cards and wait for the results (usually the next day). Since the computer was also used for college business and stuff like grades, it was not generally available to the students. If you made a mistake, you had to correct it with new punched cards and re-submit the deck. This was pretty horrible, but I found I enjoyed FORTRAN, probably because of my exposure to logic patterns. Soon after that, I learned BASIC, but the language was not as interesting.



Computer Design

I later took an EE course in computer design. The textbook was based on the DEC PDP-11 computer architecture and went into extensive detail about instruction sets, arithmetic logic units, I/O structures, and memory mapping techniques. I loved this course, even though it was all theory with no hands-on. One day I was in the lab talking to the professor and noticed a box on a shelf that said "computer breadboard" or something similar. I asked about it and he said it had never been opened and no one had looked at it. I dug into it and discovered it was a huge set of breadboard modules with transistor logic that could be assembled on an easel and wired up with jumper wires. This was like deja vu for me, but on a much larger scale than my NORDAC. It turned out to be an 8-bit serial computer which the professor let me assemble in the lab as a special project. It had 8 instructions and functioned similarly to the DEC PDP-8S, which was a cost-reduced 8-bit serial minicomputer produced in 1967. It took me a couple of days to get it working (there were many, many jumper wires) and you could single-step the clock to watch the data march through the registers. This was my first hands-on experience with something you could call a computer, although not a practical one.

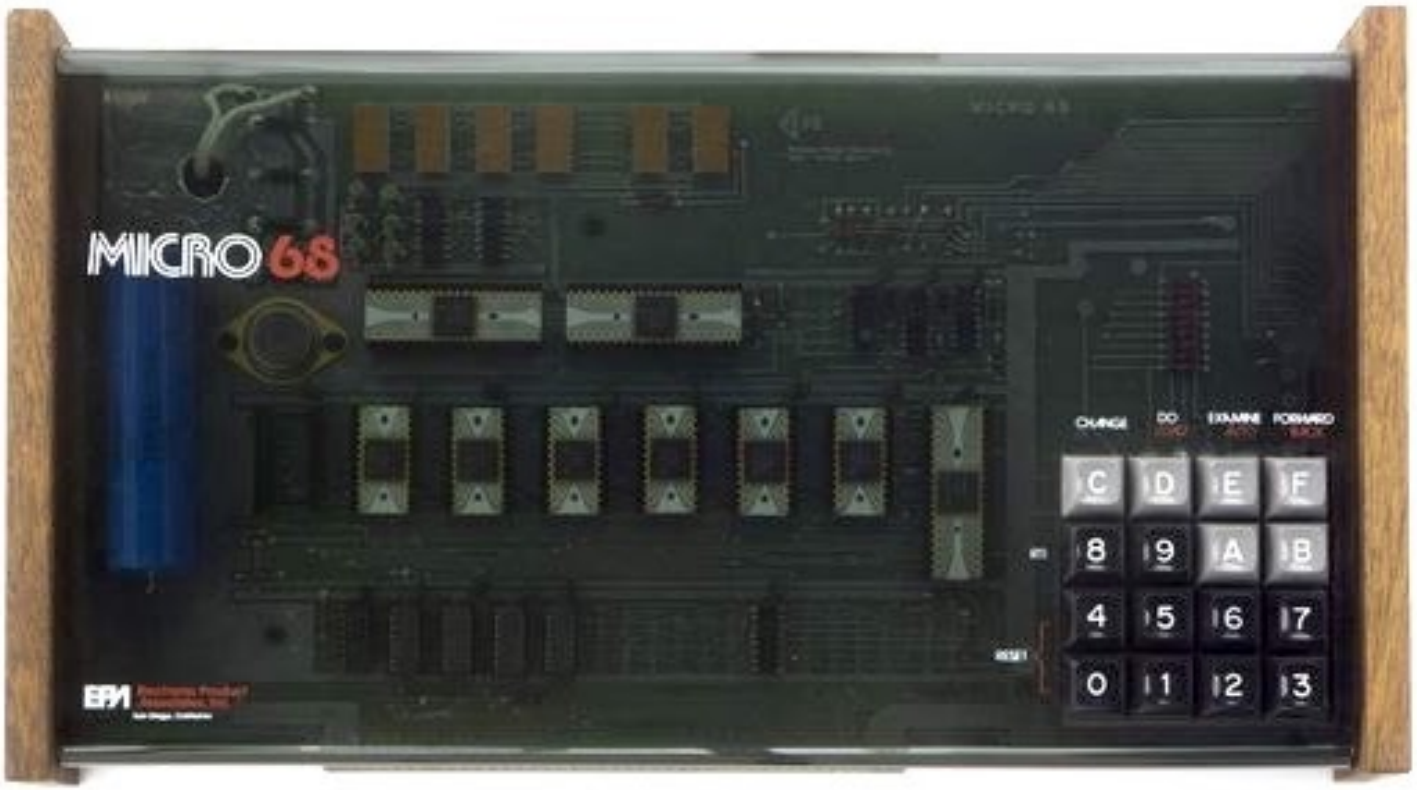
At that point, I had no idea that I would end up working at Digital Equipment Corporation for most of my career, but that's another story.

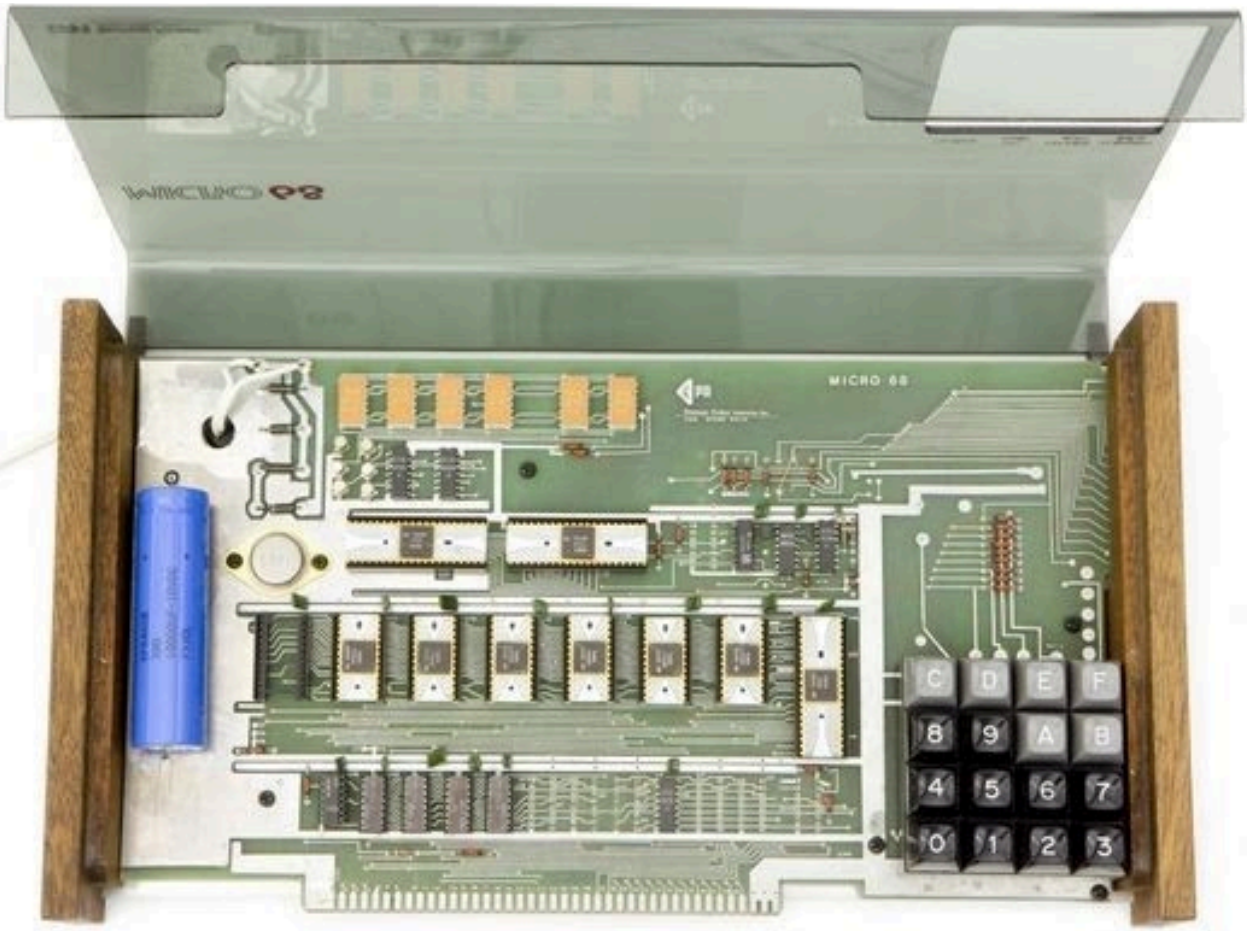
When microprocessors were a new thing, I was fascinated by the Intel 4004 chipset, originally developed for the programmable calculator market and introduced in 1971. I got the manuals for it and wanted to build a system. But things progressed rapidly, and Intel released the 8008, quickly followed by the 8080. There was an explosion of DIY computer kits such as the Altair 8800, introduced in 1974. This was soon followed by the IMSAI 8080 in 1975, pretty much a clone of the Altair. These were popularized by construction articles published in magazines like *Popular Electronics* and *Radio Electronics* and largely started the personal computer craze. These were considered hobbyist kits since they required a lot of hands-on building and low-level programming. Similarly, Apple released the Apple I in 1976, based on the low-cost MOS 6502 chip. It was not until the release of the Apple II in 1977 that a "finished" computer was available to the personal computer market at a reasonable cost. Apple was joined almost simultaneously by Radio Shack when they launched the TRS-80 the same year, based on the new Zilog Z80 chip. Both the 6502 and the Z80 were cheaper and simpler to implement at the systems level. Commodore joined in with the PET all-in-one computer, also based on the 6502, and followed up in 1980 with the lower-cost VIC-20.

I followed all these developments pretty closely but did not become an early adopter until around 1975.

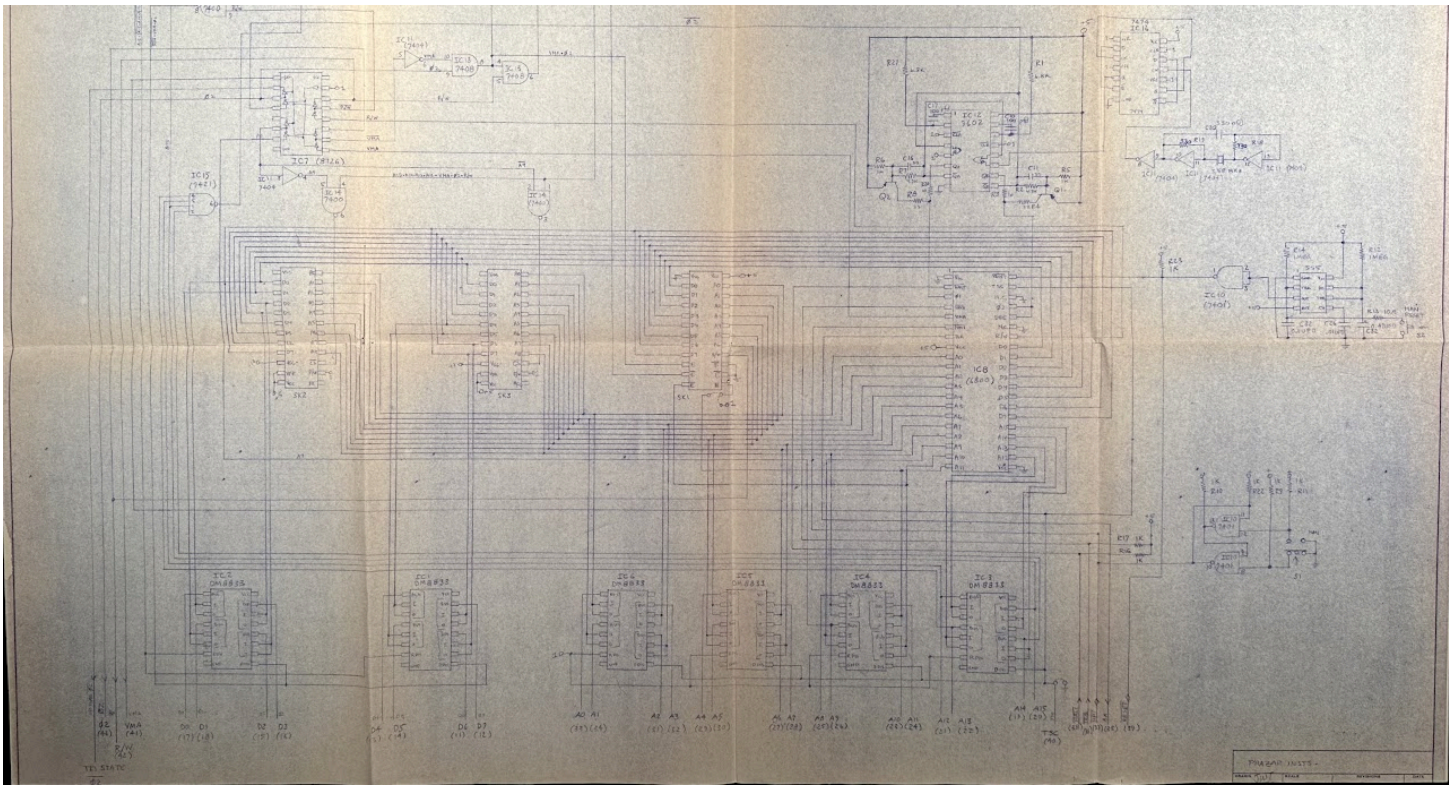
The Motorola 6800

In 1974, Motorola had introduced the 6800 microprocessor. Some of the design team revolted after its release and went off to MOS Technologies to develop what became the 6502, which had a similar architecture. To me, the 6800 had a much more straightforward architecture than the Intel chips, which seemed to have been heavily influenced by the earlier 4004 instruction set (to this day, the X86 architecture is something of a hodgepodge, but that no longer matters because the hardware is buried in layers of software). In fact, the 6800 was heavily based on the Digital Equipment Corporation's PDP-11 minicomputer architecture (which, as mentioned, I had studied in college), originally developed in the late 1960's. Since my business partner and I were looking to get into microprocessors for industrial control applications, we started looking at the 6800. Motorola published a huge applications manual that we obtained, and it was really helpful in learning how to use the microprocessor.


MICRO 68
C
D
E
F
8
9
A
B
4
5
6
7
0
1
2
3



We acquired a development board to learn how to program at the assembly level. This was a [Micro68 from Electronic Product Associates]. It had 7-segment LEDs to indicate hexadecimal addresses and data and a hex keypad for data entry. There was a monitor program in ROM to control input and display. Programming this thing in hex assembly code was extremely tedious!

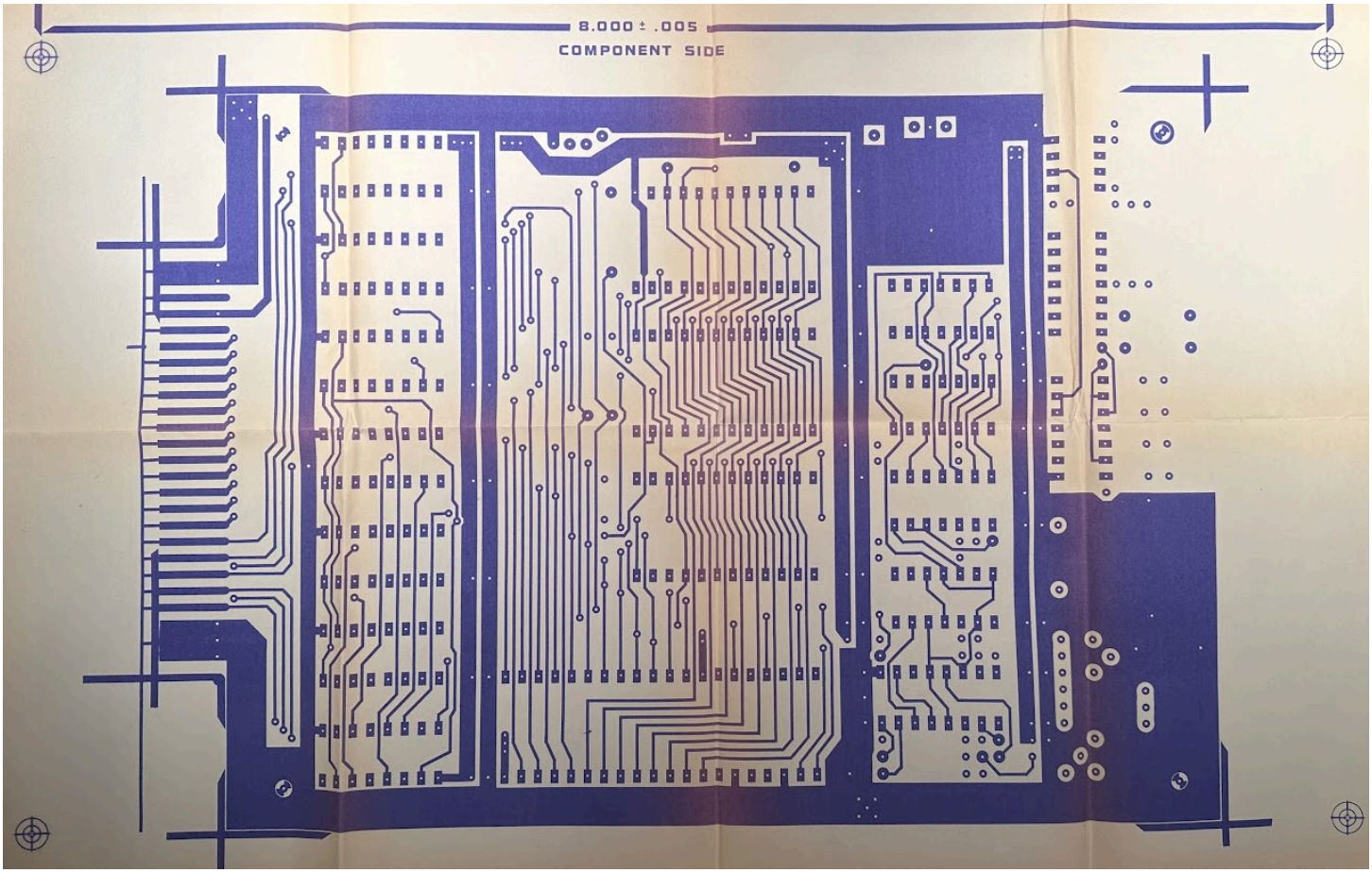


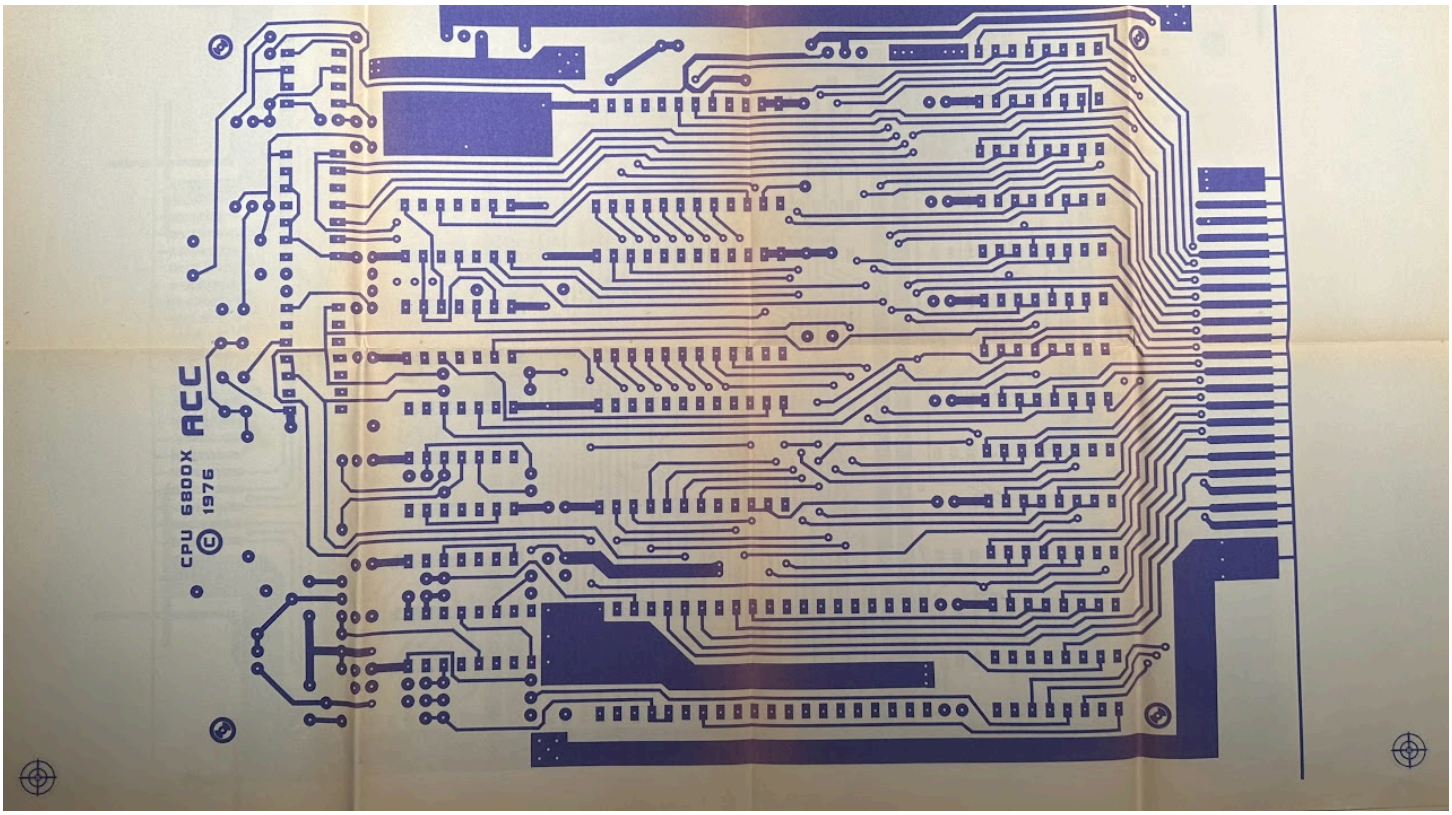
We set out to develop a set of boards to build 6800-based computers for industrial controls. Here is one of the original schematics for the CPU board I designed. These "blue line" drawings (as in blueprints) and all hand-drawn on large sheets of vellum paper.

The board had clock circuitry, bus drivers for the 8-bit I/O bus and address lines, and sockets for EPROM chips.

Ironically, no computers were used in the process of designing these computers!

with additional memory, digital I/O and serial communications, including a modem for dial-up. I still have these drawings stashed away somewhere.





Sadly, our partnership ended before we could get into production with this series of computer parts, and I went off to work for Digital Equipment Corporation (DEC) as a support engineer. However, I had learned a huge amount about microprocessors in the process. I knew there would be a solid career for me in computers, so this was a good move for me.

The TI 99/4

magazine started publication in 1975, I had become a charter subscriber to keep tabs on what was happening in the hobby. Around 1981, I acquired a TI 99/4A because it had a 16-bit processor! OK, I was a word-length snob, and 16 bits were certainly better than 8 bits, right? It had built-in BASIC so you could type in programs directly on the screen (a TV was used as a monitor). TI had a bunch of peripherals (e.g., a floppy drive) for expansion purposes, but they were rather expensive, so I never added them to the console.

The TI 99/4A

The Commodore 64

commodore
64
RESTORE
RETURN

commodore
1541

Was 16 bits important for a home computer? At that point in time, not really. 8-bit machines like the Apple II, the Radio Shack TRS-80, and lots of others were doing just fine with 8-bit processors, and it was really all about the software. So when Commodore released the C64 computer console (the 64 referred to the fact that it had 64 Kbytes of memory, quite a lot for its day), I sold the TI and bought one. The C64 became the largest-selling home computer of all time, so there were lots of software programs available. Initially, I just had the basic console, but later acquired a Commodore 1541 disk drive, making it a more like a "real" computer.

Like the TI, it had built-in BASIC as soon as it was booted up. Also, cartridges could be inserted in the rear, and it had ports for joysticks since gaming was a big deal back then.

I still have the computer and floppy drive, along with lots of software in a box, just waiting to be brought back to life. In 2014, I decided to dig it out and do just that. Here is my chronicle of that project.

So what did we do with these little machines? Games, obviously, but also rudimentary word processing and spreadsheets. Commodore had one called EasyCalc. I also added a dot-matrix printer and, for a couple of years, produced the family Christmas newsletter (for example) with crude graphics.

DOS, Windows, Getting Online

Apple II computers were selling reasonably well and they had launched the Macintosh around 1984. However, these were closed, proprietary systems and priced above what many (including me) considered a reasonable investment for home computing.

The original IBM PC (launched in 1981) architecture had an internal bus called the AT bus. Later IBM replaced it with the Micro Channel bus and peripherals were pricey due to IBM's licensing restrictions. Because the AT bus licensing was less restrictive, many clone motherboard makers adopted it and it was renamed the ISA (Industry Standard Architecture) bus, and this opened up the low-cost PC clone market. (Later, ISA was replaced by an enhanced version called EISA, standardized by a consortium of PC manufacturers).

I could see that the PC clone market was taking off, fueled by the DOS operating system, which Microsoft was making available to anyone who wanted it. I built a home system from parts (a cheap Intel 286 motherboard with ISA bus, power supply, case, modem, CDROM, etc.) acquired from a local guy and built a system to run DOS and MS Windows in 1990. I can't recall if I installed Windows 3.0 on this computer, but I definitely ran Windows 3.1 when it launched in 1992 and then later 3.11, which had local networking capability.

The company I was working for, Digital Equipment Corp (DEC), was late to the PC game, so I got up to speed on PCs and Windows on my own dime initially.

We had a 1200 (and later a 2400!) baud modem to dial into America Online (AOL - You've Got Mail). I was a charter subscriber to AOL for DOS in 1991 since internet over cable (broadband) was not available to us until about 1994. AOL had previously launched for Apple and Commodore 64, but I passed on it. AOL had a brilliant marketing strategy - they would mass-mail free CDs with their software to everyone and offered a free trial period. Some estimate that over a billion AOL CDs were mailed out (many people received multiple copies over time). Other providers like Prodigy and CompuServe were more expensive then so AOL became pretty popular.

As it was, AOL was very limited - you could do email and they had bulletin boards and news, but since the internet was in its infancy, there was not much online content. When Netscape launched the first WIndows-based web browser in 1994, things started to take off.

It is hard to believe, but back then this was the state of connectivity.

gotten around to making IBM compatibles, so I had access to them). Neither was software, since I had an MSDN license courtesy of DEC and access to all the software Microsoft produced. Nowadays, we use MacBooks and iPhones at home because, well, they just work.

But - Wait for it - I recently (2023) decided I needed to have a Windows computer back in my shop, so see my article on a cheap Windows computer.

The Retro Computing Hobby

The have been several attempts to resurrect or emulate old machines. I'll detail a few here, including the PiDP-11 and PiDP-10 kits that I have built.

Recently, there seems to be a great deal of interest in simpler, Transistor-Transistor Logic (TTL) based designs. These are minimalistic designs intended to be educational and can be used as teaching tools.

name of Marcel van Kervinck, working with another engineer named Walter Belgers. The project was named Gigatron and is described here.

What made this design unique is that it did not use a microprocessor at all or even an ALU chip, but instead was completely implemented in standard 7400-series TTL logic chips (the first of which were available around 1970), and it only used 33 of them. It was micro-programmed with logic stored in a ROM. The advantage for people learning about computers is that it is completely understandable and not opaque like a single-chip microprocessor. Even more remarkable is that it is a full 8-bit color computer that can be built for under $200 and can emulate the Apple I.

The Gigatron Single Board Computer all in TTL

Ironically, this design could have been implemented before Wozniak designed the Apple I around the 6502 chip. This design has even fewer parts! To be fair, the 32K static RAM chip used In the Gigatron was not available in 1976 - and the largest was 4K - so the design would have been expensive to implement back then. In reality, all RAM was pricey at that time. Woznok used the first available 4K dynamic RAM chips to keep the cost down. Regardless, people were so enamored with using microprocessors, they overlooked simpler approaches like this. DEC had created (nearly) single-board TTL computers like the PDP11/05 in the early-mid-1970s, but they did use bit-slice ALUs. This was followed by the PDP11/04 (a true single-board computer minus memory). One of my co-workers at DEC gave me the original wire-wrapped prototype of the PDP11/04 (he was directly involved with the project) on a hex-sized module, which I hung on my office wall as a piece of techno-art. It probably had about 200 ICs, but bear in mind the PDP11 h a very sophisticated architecture and complex bus structures (the Unibus).

you can get the PCB Gerber files from the Gigatron website and get your own boards made. Plus they have a complete BOM of parts. I think this would be ideal for a club or high school computer course.

Here is a description from the Gigatron website:

> *The Gigatron TTL microcomputer is a minimalistic retro computer. It's special in its own oddball way, because it has absolutely no complex logic chips in it, not even a microprocessor! Its CPU is built out of a handful of classic 7400-series ICs, colloquially known as the TTL logic series. These chips combined form a powerful 8-bit processor.*
> *Besides running applications, this processor performs functions that traditionally require dedicated peripheral chips for video, sound and other I/O. By eliminating these the hardware remains small and understandable. Still the single-board system works as a full-blown microcomputer that you can play video games with.*
> *Now you can easily build one yourself. As you build, learn what happens inside a CPU by looking inside one. See the functional units, look inside the Arithmetic and Logic Unit, see its truth tables, and learn what makes up a ROM. Then go on to enjoy playing the built-in retro video games or write little programs in BASIC. You can also hack it in any way you like if you have a taste for that.*

Ben Eater

Another experimenter who falls into the minimalist-retro computing category is Ben Eater. His website describes several designs and breadboards for "simple" projects such as an all-TTL 8-bit computer. This computer was based on a design by Albert Paul Malvino, who wrote a book called *Digital Computer Electronics**. In it, he described a simple computer called the SAP1 (Simple As Possible), Eater also has a project to build a 6502 chip based design, but he recommends the TTL design for learning how computers work at a much lower level. He also covers the basics, like simple logic circuits and how adders work. He provides schematics and kits and has a great YouTube channel. Ben's explanations are crystal-clear and he is very humble in his teaching approach - it works for all audiences.

*This text has a lot more in it than just the SAP1 - it is a very complete survey of computer architectures where the author develops more advanced computer designs and has a comparison of early microprocessors as well as programming techniques at the machine code level. A worthwhile read for anyone interested in how computers really work.

videos explaining how Turing machines work and how to code them, as well as complete designs, one of which will emulate the Apple][. You can learn a lot from his channel. He took the Ben Eater SAP1 design and made it even simpler, calling it the Turing SAP1. His approach is to use larger EPROMS to replace as much combinatorial logic as possible with firmware, including ALU functions, reducing the total chip TTL count to 17. This is basically a finite-state machine with registers used to hold data during state changes. He draws the schematics with KiCad in real-time (in itself a good learning experience) and wires up breadboards as he is speaking in the videos so you can follow along. His approach is not as straightforward as Eater's but is worth studying.

Regan has a second series of videos where he extends the functionality of the SAP1 into a (more complex) 6502 TTL computer capable of running Apple][software (games, etc.).

Hackaday Homebrew Projects

There is a lot of interest on Hackaday in creating small projects to build simple processors and systems. See this link for some of them. One person is even building a CPU with LED-Transistor logic gates. Other designers are using techniques like Field-Programmable Logic Arrays (FPLA's) to implement CPU functions.

Computer Museums and Festivals

There are a number of computer museums around the world dedicated to preserving the knowledge and history of the computing arts. A fairly complete list is here on Wikipedia. Not too long ago, very few existed, but DEC had one at its Marlboro, Mass facility. That was later moved to downtown Boston, and then relocated to Mountain View, CA, where it was combined with a much larger collections from various manufacturers. There is also one in Rhode Island and one in Maryland. Generally, these are staffed by volunteers who attempt to resurrect old machines and get them back to working condition.

David Lovett is a well known YouTuber, under the name of Usagi Electric, who restores ancient computers. He is even restoring an old Bendix (yes they made computers once upon a time) tube-based computer for one of the museums. He also has built a single bit (serial) computer entirely out of vacuum tubes.

The largest regular gathering of retro computing enthusiasts is called the Vintage Computer Festival (VCF). This draws hundreds of attendees, including David, and features demos of restored machines and software. Also - Oscar Vermeulen (see below) has been instrumental in reviving interest in old DEC machines with his Raspberry Pi based simulators and goes to these events. The most recent schedule is here.

My Recent Projects

The DEC PDP 11 Emulator - PiDP 11/70

To bring things full circle for me at least, a few years ago, I learned of a project to produce a kit that replicated the DEC PDP 11/70 from the 1970s. The project was documented here and brought to life by a guy named Oscar Vermeulen. Oscar had previously made a kit that emulated the DEC PDP 8 based on a Raspberry Pi running simulation software, and it had become very popular. When he announced the PiDP-11 (the name incorporated Pi) I was first in line to order one. In my early days at DEC, I had cut my teeth on PDP-11s, and the 11/70 was my favorite machine. This kit is a 2/3 scale replica of the PDP11 front panel, complete with working blinking lights and switches that function identically to the original.

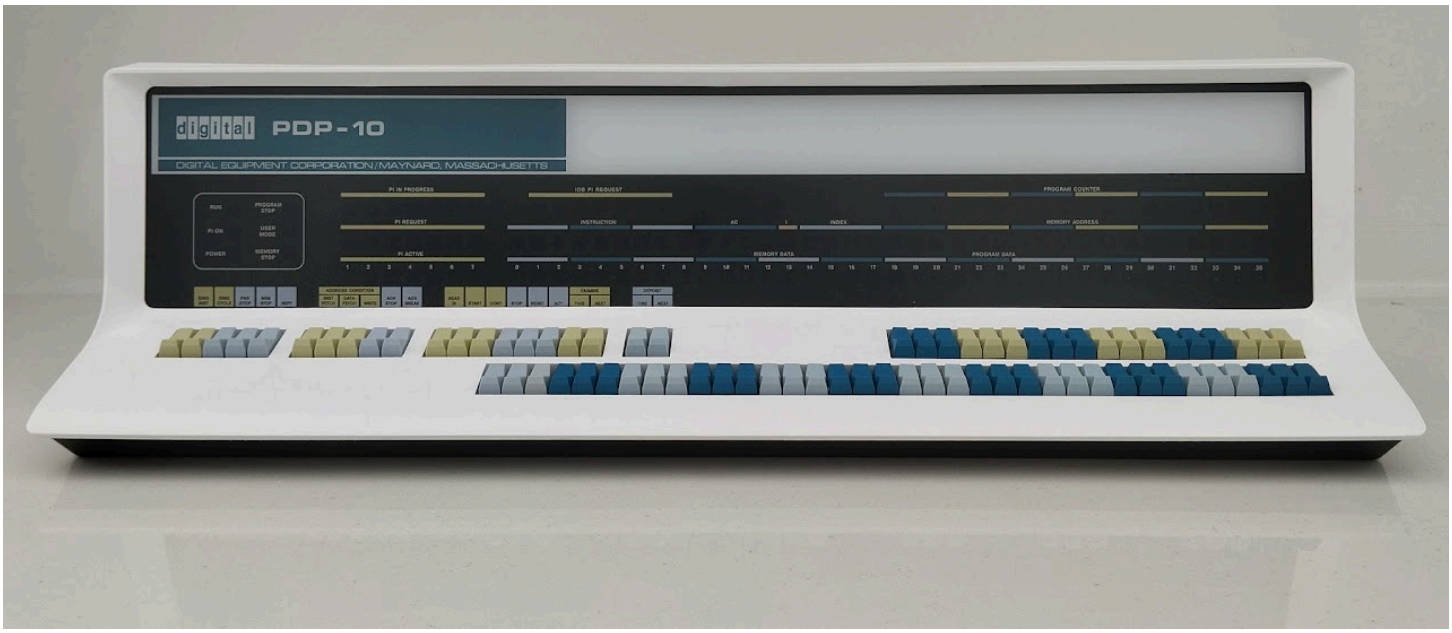
for a VAX CMOS chip project that obsoleted the enormously expensive VAX 9000 mainframe project less than a year after its introduction (that's a whole 'nother story). SimH is an open-source project that has been enhanced over the years by others to allow the emulation of all sorts of computers, mostly obsolete today (even the Altair).

In the case of the PiDP-11/70, there is a whole host of legacy operating systems (available in multiple versions) that can be booted up. These include:

- DEC RSTS
- DEC RSX11
- DEC RT11
- BSD UNIX
- SYSTEM V UNIX
- ULTRIX UNIX
- XXDP diagnostics
- and others

Each of these has programming languages, games, and utility programs, so it is an endless playground for retro computing idiots like me.

The DEC PDP 10 Emulator



Here is a photo of his prototype panel - a beautiful molded panel and very authentic looking - just 2/3 scale.

Oscar Vermeulen had also been working on a kit that simulates the DEC PDP-10 mainframe, complete with fully functional blinking lights and switches. It can run TOPS-10, the original large-scale DEC timesharing OS, as well as ITS, the somewhat famous (but not well-known) Incompatible Timesharing System developed at MIT as a completely open system. TOPS-10 by itself would be kind of boring, but ITS opens up a whole set of applications and games that MIT hackers created. The PiDP-10 team has resurrected many of the original ITS files and tested them on the PiDP-10.

The simulator has 124 LEDs and 74 switches, all interfaced with the Raspberry Pi GPIO bus.

Building the PiDP-10

I completed the build of my kit and added this page to document some tips I came up with in addition to Oscar's very good instructions.

time-sharing industry was ramping up circa 1966, and thousands were sold to companies such as Prodigy and CompuServe, as well as many universities. The DEC operating system, TOPS 10, was strictly aimed at text-based timesharing, initially supporting teletype terminals. Many programming languages eventually became available, making it extremely useful for software developers:

- **FORTRAN** (Versions like F40 and FORTRAN-10 for numerical and scientific computing)
- **COBOL** (COBOL-68 for business and administrative applications)
- **BASIC** (Interactive, conversational language for general use)
- **ALGOL** (ALGOL-60 for algorithm development and academic computing)
- **MACRO-10** (Assembly language for low-level machine programming)
- **BLISS** (DEC's proprietary systems programming language)
- **APL** (APL-SF for mathematical modeling and array manipulation)
- **FOCAL** (DEC's proprietary conversational interpreter)
- **AID / JOSS** (Algebraic Interpretive Dialogue for time-shared mathematical calculations)
- **IQL** (Interactive Query Language used alongside the DBMS-10 Database Management System)

Famously, Bill Gates and Paul Allen first learned to code on a DEC PDP-10 time-sharing computer while in middle school. Years later at Harvard University, they used a campus PDP-10 to write Altair BASIC. This BASIC interpreter was Microsoft's very first product. You could say that the DEC PDP-10 was at the core of Microsoft's early success, and even largely responsible for launching the software industry. Early models of the PDP-10 could support 25-50 simultaneous on line users, depending on workload.

But other researchers had a slightly different idea.....

At MIT, there had been a long series of computing projects involving some of the early post-WWII computer architectures and software developments, most notably beginning with the Whirlwind tube-based computer. Next, the TX-0 and TX-2 computers were designed and built at MIT's Lincoln Labs. These were among the first all-transistor machines, replacing vacuum tubes. One of the engineers on the project, Ken Olsen (an MIT grad), saw an opportunity to start his own company and in 1957 Digital Equipment Corporation (DEC) was launched. Initially, the company built simple electronic modules for the scientific and research markets. But Olsen had a goal of building a computing machine consisting of many small modules and using the knowledge he had gained at MIT, so he directed the building of the DEC PDP-1 (see further down this page for info on the PiDP-1 simulation project). This was the very first "personal" computer. It had a graphics terminal and users could interact with the screen in real time. Olsen donated a PDP-1 to MIT in 1961 and it quickly became a favorite of the "hacker" culture. Some of the very first video games such as "*Spacewar!*" were developed on the PDP-1 there.

MIT also created what was referred to as Project MAC (for Mathematics and Computation) which looking at creating (among other things) a timesharing system running on an IBM 7094. This became known as the Compatible Time Sharing System (CTSS). Around the same time, the

inspiration for Unix, a more modern implementation, developed around 1969 at Bell Labs using DEC machines.

Meanwhile, at DEC, the PDP-1 was quickly superseded by a series of newer DEC computers. The first large-scale 36-bit machine was the PDP-6, introduced in 1964. DEC only sold a handful of these but one ended up at the MIT AI lab. A small group of hackers was unhappy with CTSS and wanted a more "open" system. They developed a new OS, with the tongue-in-cheek name of Incompatible Timesharing System (ITS) which they wrote for the PDP-6 in machine assembly language and released it in 1967 for use by AI programmers at MIT. This eventually led to a split between the AI and MAC groups. The MAC group continued to focus on CTSS and MULTICS but was populated by people with ties to large corporations (like IBM and GE), which was repellent to the hacker-focused AI group. For a good history of the MIT groups and the evolution of these computer technologies, see this paper.

One of the most significant features of ITS was that it had zero security, by design. Users could look at and edit each others' files and projects and did not even have to log in, although it was considered polite. There was a "turist" login username for lab visitors. The group added all sorts of peripherals to the machine, notably graphics terminals such as vector and raster displays. Even the first laser printers were supported. The system was also one of the earliest connected to the ARPAnet. It even provided users with an instant messaging capability. All these features were way ahead of their time. Note that this preceded the famous Xerox Alto project by several years.

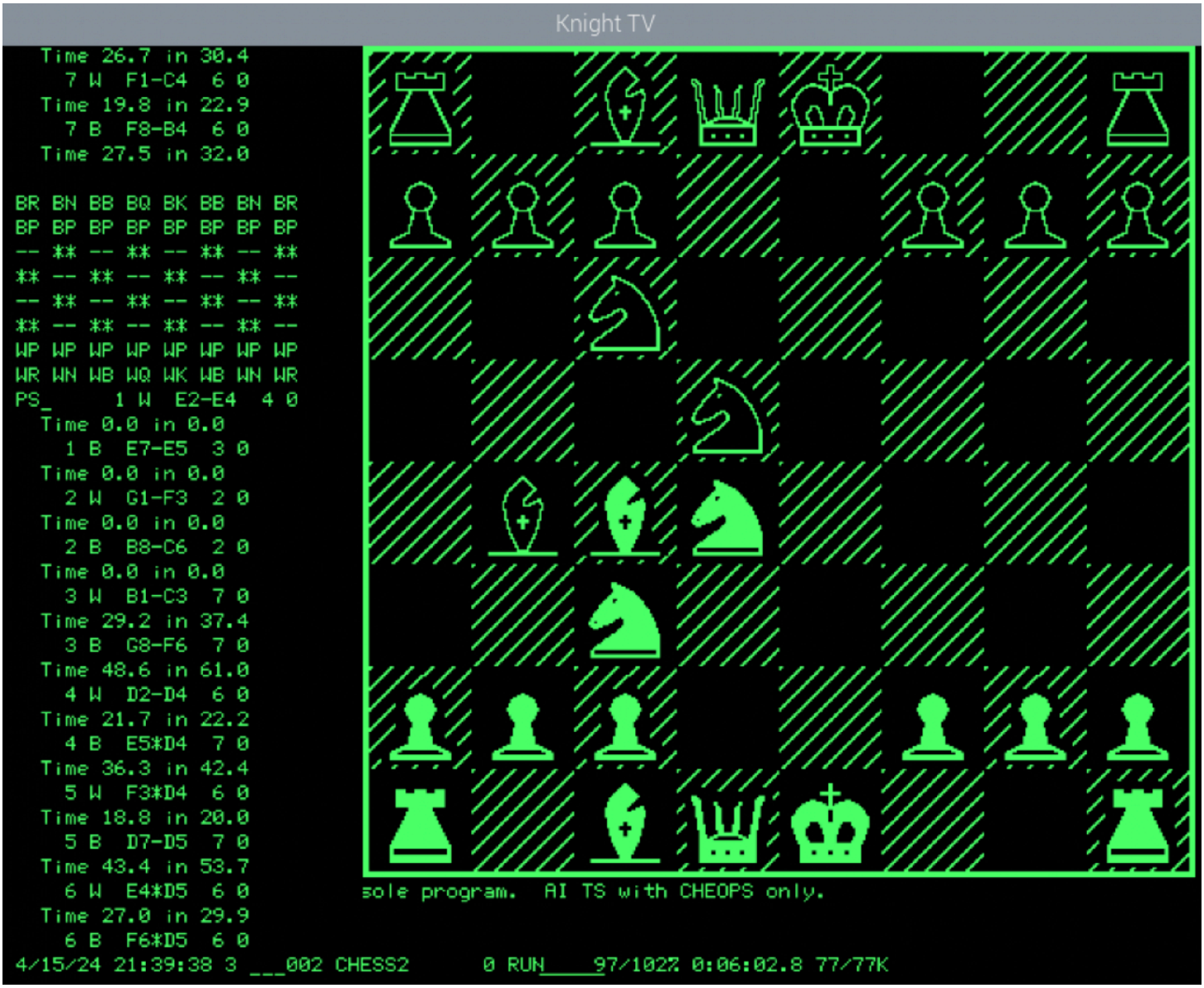
In 1966, DEC released the 36-bit PDP-10, which was mostly compatible with the PDP-6 (but much more scaleable and reliable) and one was delivered to the AI lab. The ITS group ported their OS to the PDP-10 and even combined the PDP-6 in a shared memory configuration, so the installation became one of the first multi-processor computers! One of the developers, Tom Knight, created a terminal known as Knight TV, which was made available to multiple users. The TV system used a DEC PDP-11/20 (introduced in 1970) which had a Unibus backbone that was interfaced directly to the PDP-10, further extending the idea of a closely-coupled multi-processor system. Knight built a video switch and keyboard multiplexers to interface the terminals to the PDP-11 and Knight TV became the preferred method of interacting with ITS.

ITS spawned the creation of significant software projects, including the EMACS editor, the LISP programming language, the game Zork, Mac Hack (the leading graphical computer chess program for many years), and many others. More importantly, it became the foundation for open systems in general.

The ITS project remained in use until about 1990, so it had a long life. Interest in the OS was continued at the Stacken Computer Club in Sweden until about 1995. Lars Brinkhoff has been instrumental in resurrecting many of the ITS projects and collaborating with Oscar Vermeulen on the PiDP-10 project, along with Rich Cornwell who is working on the SimH-based emulator for the KA-10 processor (the original PDP-10 CPU).

Some history of the DEC PDP-10 and ITS is intertwined in a very interesting way. And now with new emulator projects and the resurrection of the original software projects, that history is being brought

operating the system. I suspect that the "hackers" felt obligated to document the system since a lot of the funding was provided by ARPA.


Knight TV
Time 26.7 in 30.4
7 W F1-C4 6 0
Time 19.8 in 22.9
7 B F8-B4 6 0
Time 27.5 in 32.0
BR BN BB BQ BK BB BN BR
BP BP BP BP BP BP BP BP
-- ** -- ** -- ** -- **
** -- ** -- ** -- ** --
-- ** -- ** -- ** -- **
** -- ** -- ** -- ** --
WP WP WP WP WP WP WP WP
WR WN WB WQ WK WB WN WR
PS_ 1 W E2-E4 4 0
Time 0.0 in 0.0
1 B E7-E5 3 0
Time 0.0 in 0.0
2 W G1-F3 2 0
Time 0.0 in 0.0
2 B B8-C6 2 0
Time 0.0 in 0.0
3 W B1-C3 7 0
Time 29.2 in 37.4
3 B G8-F6 7 0
Time 48.6 in 61.0
4 W D2-D4 6 0
Time 21.7 in 22.2
4 B E5*D4 7 0
Time 36.3 in 42.4
5 W F3*D4 6 0
Time 18.8 in 20.0
5 B D7-D5 7 0
Time 43.4 in 53.7
6 W E4*D5 6 0
Time 27.0 in 29.9
6 B F6*D5 6 0
sole program. AI TS with CHEOPS only.
4/15/24 21:39:38 3 ___002 CHESS2 0 RUN 97/102% 0:06:02.8 77/77K

Here is a screenshot of the ITS Chess2 program running in the simulated Knight TV terminal environment. The simulation also emulates the PDP-11/20 controller connected to the simulated PDP-10. using the software provided by the PiDP-10 project.

This is all running on my Pi 5 and the display is handled remotely by VNC Viewer on a Mac. I installed the project software in anticipation of getting the new PiDP-10 console hardware, but it runs fine in a stand-alone environment on the Pi. I finally got and built the kit as mentioned above and documented the build here.

Bear in mind that this software was written in about 1970, over 50 years ago!

Angelo Papenhoff has compiled a nice website with links documenting some of the history of the PDP-6 and PDP-10, along with some info about the resurrection of Knight TV. He and Lars Brinkhoff collaborated on getting the Knight TV emulation package working with ITS.

A copy of the *Digital PDP-10 System Reference Manual* can be found here. This manual mainly covers the impressively large (for 1966) instruction set in assembly. A copy of the *PDP-10 KA10 Central Processor Maintenance Manual* - focused on the CPU hardware is here.

Oscar's GitHub repository is here. Check here for the latest versions of the PiDP software.

Oscar's PiDP-10 KiCad files with schematics and PCB layouts can be downloaded here.

Andrew Barron has published a book about the PiDP-10. It is available here on Amazon.

RetroBytes has published a pretty cool YouTube video about the PiDP-10 here.

A more recent project from Oscar V. and team is a replica of the 18-bit DEC PDP-1, the first computer produced by Digital Equipment. This was launched in 1959, only two years after Ken Olsen founded the company. It was a revolutionary product, the first fully interactive "personal" computer for the commercial market, although modified versions of it were used as time-sharing machines. Many basic design principles were carried over from the MIT Lincoln Labs TX-2 fully transistorized computer, which Olsen collaborated on.

The website for Oscar's project is here. Also, there is an online manual here.

I got another Raspberry Pi 5 and loaded the software to try out. There is a web UI version which allows you to visualize the panel and the paper tape reader/punch. At some point, I will get the kit (as time allows).

Members of the Google interest group have added various extensions to the emulation, including a drum storage simulator and a reproduction of the famous DEC Type 30 display, scaled down to match the PiDP-1.

Several people have commented that the PDP-1 is actually more "fun" than other simulations. This is for various reasons, but the main one seems to be its simplicity and approachability. Since it is considered the first graphics-enabled computer, programming graphics demos and games is fairly straightforward and understandable.

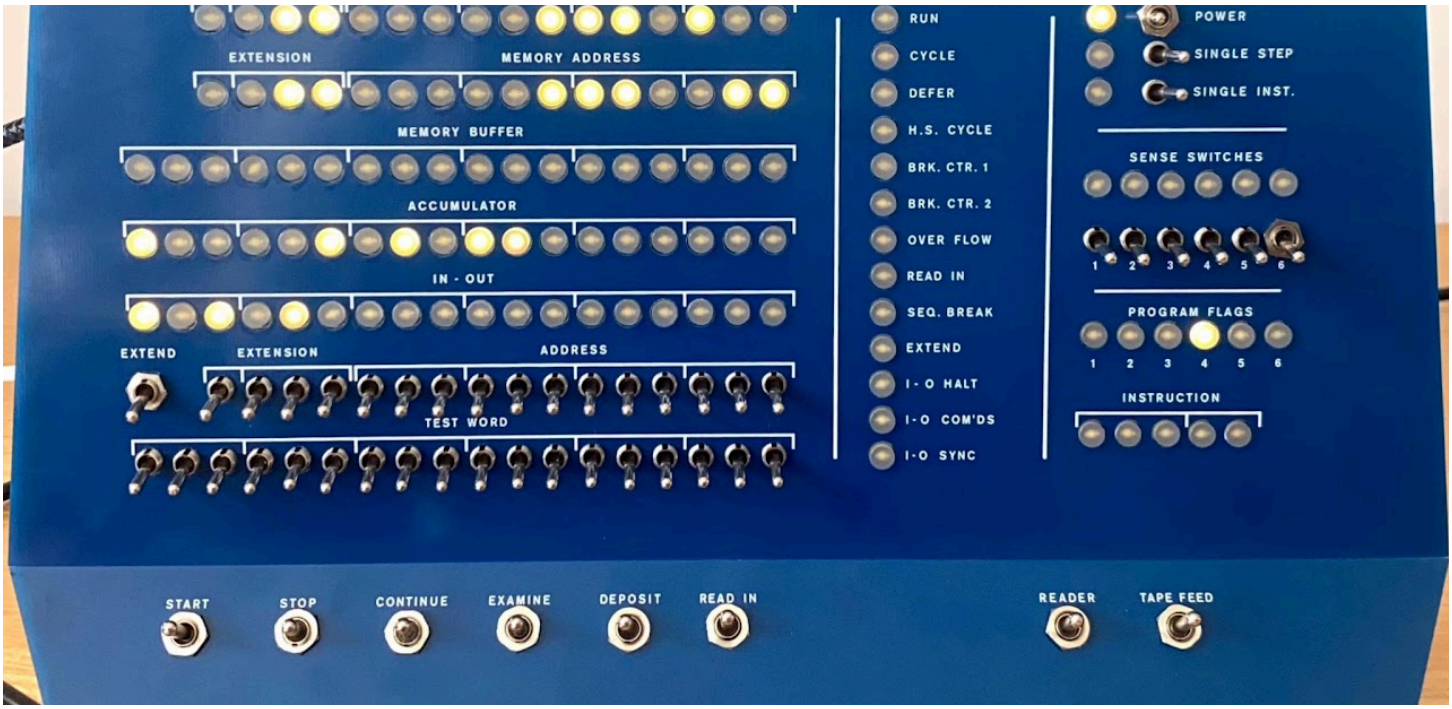
Update June 2026

I could not resist, so I finally ordered a kit. A build blog will follow soon. There are two versions of the kit: a table top console and a larger wall-mounted version (like a picture frame). I ordered the console version.

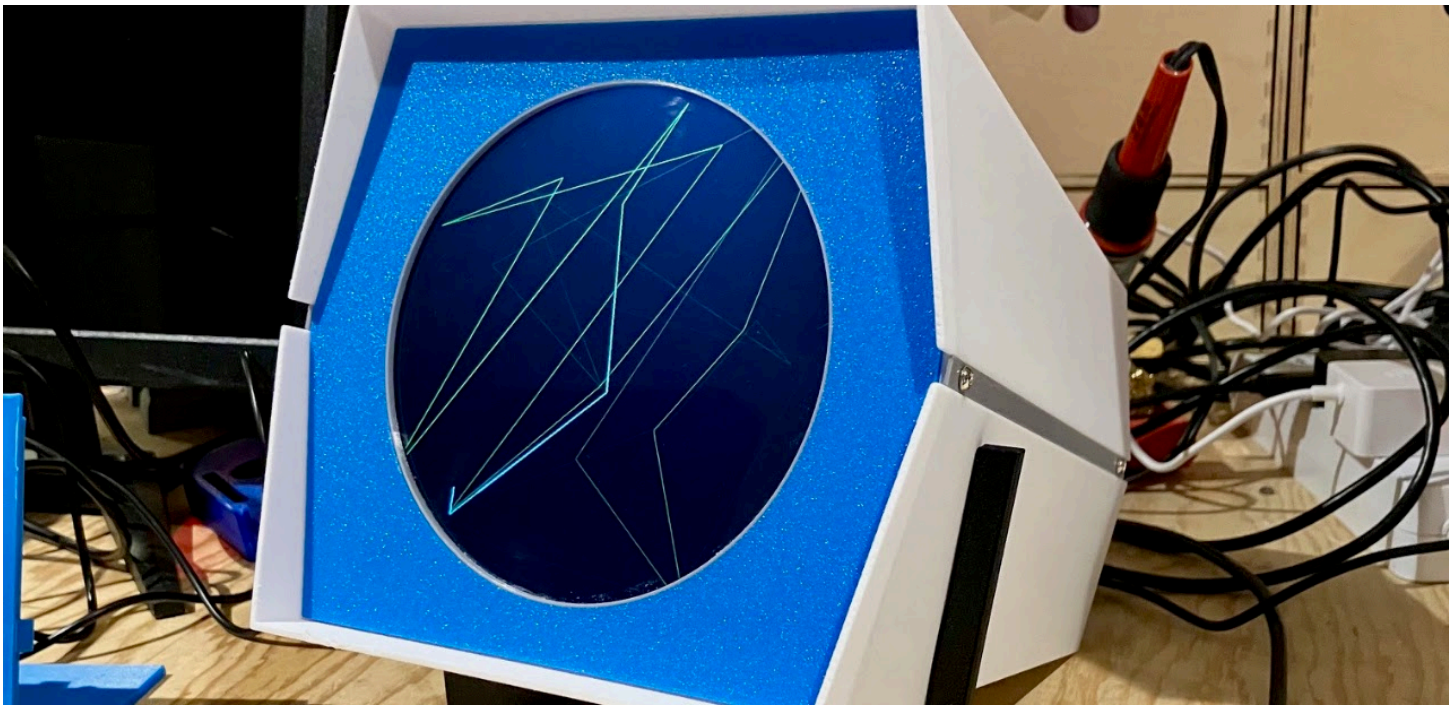
July 2026

My kit arrived from Panama in record time. I ordered it on a Monday and received it on Friday via DHL.

Here is a new page dedicated to my PiDP-1 build.



Here is the panel of Oscar's reproduction kit. Another example of a faithful blinkin lights and switches simulation. As with the other kits, the switches and lights are all fully functional and behave exactly as the original.



Michael Gardi has posted a project on Hackaday that describes this reproduction of a 36% scale DEC Type 30 vector display. This display was what originally created excitement among early hackers who got access to a DEC PDP-1.

Why Blinkinlights?

If you are baffled about the interest in these old simulations with lots of switches and lights, read this article where I try to explain the fascination.

Copyright 2026 - JWT Audio